CPSC 1155 – Assignment 2

Selection Statements / Random Numbers

Objectives

The goal of this assignment is to develop algorithms and C++ programs using selections, and random numbers.

Readings

You should be reading Chapter 3 of the textbook. The lectures and labs will provide additional supporting material.

Instructions

For each of the following problem statements:

1. Read the problem statement and clarify the problem.
    a. Break the problem into smaller problems if needed.
2. Determine the IPO.
    a. Determine input, output, constants, and conditions.
    b. Declare the variables and constants (data type + meaningful names).
    c. Work out the problem by hand using typical input values. Determine the range of valid input values.
    d. Determine the process.
3. Write a pseudocode as required.
4. Write a C++ program (use the given filename) that implements the pseudocode (or algorithm).
    a. Add comments where needed. Make sure to use a comments header to reflect the intention of your program and name of the author (you) and the date the program was written.
    b. Test, debug, and execute the program using typical values.

Submit according to the instruction in the "Submission" section.

Problem Statements

1. (is_right_side_triangle.cpp) Write a **pseudocode** and **C++ program** that reads three positive integers for the edges of a triangle and determines if it is a right triangle. If it is, the program displays that the triangle is a right triangle. Otherwise, the program displays that the triangle is not a right triangle. The program does not accept negative values.

   Hint: A triangle is a right triangle if one angle is 90° degrees or $\frac{\pi}{2}$ radian (What is cos (90°)?). We can use the law of cosines to find the angles. Note that the order of a, b, and c is important in the formula.

$$\cos(A) = \frac{b^2 + c^2 - a^2}{2bc}$$

   Here are two sample runs:

```
Please enter lengths of a triangle: 3 4 5
The triangle with lengths 3 4 5 is a right-side triangle
```

```
Please enter lengths of a triangle: 4 5 6
The triangle with lengths 4 5 6 is a not a right-side triangle
```

```
Please enter lengths of a triangle: -3 4 5
The values must be positive
```

2. (simple_caculator.cpp) Write a **C++ program** that asks the user to enter two integers and an arithmetic operator (+, -, *, /). Then based on the operator, the program calculates the result. The program checks input for invalid operators. It also checks for division by 0.

   You may use a char data type for the operator. A character literal is enclosed in single quotation marks. For example:

   ```
   char operator = '+';
   ```

   Here are sample runs:

   ```
   Enter two integers: 4 8
   Enter an arithmetic operator (+, -, *, /): +
   Result = 4 + 8 = 12
   ```

   ```
   Enter two integers: 4 8
   Enter an arithmetic operator (+, -, *, /): $
   Invalid operator.
   ```

   ```
   Enter two integers: 4 0
   Enter an arithmetic operator (+, -, *, /): /
   Cannot divide by 0.
   ```

3. (sort_three_integers.cpp) Write a **C++ program** that prompts the user to enter three integers and displays the integers in decreasing order.

   Requirement: Compare every two integers and swap them if needed. Three swaps might be needed.

   Here is a sample run:

   ```
   Enter three integers: 1 2 3
   Sorted in descending order: 3 2 1
   ```

4. (find_future_dates.cpp) Write a **C++ program** that prompts the user to enter an integer (from 0 to 6) for today's day of the week (Sunday is 0, Monday is 1, . . . , and Saturday is 6) and another integer (any positive integer) for the number of days after today to show a future day.

   c. Use variables to store current day and future day.

   d. Use switch to find out what day of the week is today.

   e. Use if-else if-else to find out what day of the week is the future day.

   f. Check that the first input is between 0 and 6 inclusively, and the second input is positive. If they are invalid, the program terminates and prints error messages. See below for the error messages.

   Here are the sample runs:

   ```
   Enter today's day: 1
   Enter the number of days elapsed since today: 10
   Today is Monday and the future day is Thursday

   Enter today's day: 10
   Enter the number of days elapsed since today:10
   Invalid today's day, it should be between 0-6 inclusively

   Enter today's day: 1
   Enter the number of days elapsed since today: -1
   Invalid number of days elapsed, it must be positive
   ```

5. (palindrome.cpp) A palindrome is a number or a text phrase that reads the same backward as forward. For example, each of the following six-digit integers is a palindrome: 123321, 555555, 455554, and 116611.

   Write a **pseudocode** and **C++ program** that reads in a six-digit integer and determines whether or not it is a palindrome. You must use the division and remainder operators. Assume input is correct (six-digit integer).

6. (two_circles.cpp)[i] Write a **C++ program** that prompts the user to enter the center coordinates and radii of two circles and determines whether the second circle is inside the first or overlaps with the first. Inputs are floating point numbers. The program checks that the input radii are not negative. The user input order should be x1, y1, r1, x2, y2, r2.

   a. $circle2$ is inside $circle1$ if $the\ distance\ between\ the\ two\ centers \leq |r1 - r2|$
   b. $circle2$ overlaps $circle1$ if $the\ distance\ between\ the\ two\ centers\ \leq r1 + r2$

   Depending on the inputs, one of the following outputs should be shown:

   - "Radii must be positive"
   - "circle2 is inside circle1"
   - "circle1 is inside circle2"
   - "circle2 overlaps circle1"
   - "circle2 does not overlap circle1"

7. (pick_card.cpp) Write a **C++ program** that simulates picking a card from a deck of 52 cards. Your program should display the rank (Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, King) and suit (Clubs, Diamonds, Hearts, Spades) of the card. You should use random numbers, so each time you run the program, the result should be random. Here is a sample run:

```
The card you played is Jack of Hearts
```

8. (bandit.cpp) Write a **C++ program** that simulated an encounter in an RPG game. You are a lone traveler in the woods. Suddenly, a bandit appears in front of you demanding you to give him all your money.

   You have the following choices:

   1. Try to flee.
   2. Fight!
   3. Try to persuade him that robbery is bad, and he is better than that.

   Input 1, 2, or 3 to make the choice. Check that input is valid (1, 2, or 3). Now simulate the outcome. Let's flee has 20% chance of success, fight has 50% chance of winning, and persuasion has 5% chance of success. Display the outcome of the encounter. Make a separate outcome message for each possible outcome. The exact messages are up to you. The only requirement is that they are descriptive.

   Here is a sample run:

```
You are a lone traveler in the woods. Suddenly, a bandit appears in front of you demanding
you to give him all your money. You have the following choices:
1. Try to flee.
2. Fight!
3. Try to persuade him that robbery is bad, and he is better than that.
Please enter your choice (1,2,3): 2
You draw out your sword and fought! It was a close one, but you managed to defeat the
bandit!
```

   Have fun with this one!

Submission

Submit a zip folder named as yourName_Assign2.zip to Brightspace. This folder should consist of **C++ codes** in individual .cpp files and one pseudocode.txt file with all your **pseudocode** (for questions 1 and 5). Make sure to name each .cpp file exactly as the instructed names at the beginning of each question.

Make sure that all your .cpp files compile and run properly before submission. Your file must run properly in order to receive full marks.

Marking Scheme

There are 10 marks for each question with the following details:

Question 1: is_right_side_triangle.cpp

- 5 for pseudocode
- 5 for correct solution

Question 2: simple_calculator.cpp

- 2 each for calculating typical + - * / cases
- 1 for checking invalid operator
- 1 for checking division by 0

Question 3: sort_three_integers.cpp

- Must use swap

Question 4: find_future_dates.cpp

- 2 correct solution with typical input
- 3 for using switch statement
- 3 for using if else statement
- 2 for checking inputs

Question 5: palindrome.cpp

- 5 for pseudocode
- 5 for correct solution

Question 6: two_circles.cpp

- 8 for correct solution
- 2 for checking that radii are positive

Question 7: pick_card.cpp

- 10 for correct solution

Question 8: bandit.cpp

- 9 for a solution that displays all possible outcomes with correct percentages
- 1 for checking valid input

---

[i] This question is taken from Liang's text book Exercise 3.26.